IME I PREZIME:_____ Ak. god. 2022./2023.

JMBAG:_____

1. domaća zadaća iz Formalne verifikacije programske potpore

NuSMV

Najprije je potrebno instalirati sustav NuSMV prema uputama u datoteci "FVPP_NuSMV_upute.pdf"

1. dio

1.1. Proučite primjer **mutex_1ex.smv**.

1.2. Specificirajte i napišite u CTL notaciji obilježje sigurnosti (engl. *safety property*):

«Dva procesa ne mogu biti istovremeno u kritičnom odsječku.»

Potrebno je napisati dva oblika obilježja:

a) specifikacija da je moguće jedno nepoželjno ponašanje (engl. *refutation*)

b) specifikacija da nema nepoželjnog ponašanja

Nepoželjno ponašanje je u ovom slučaju istovremeno nalaženje u kritičnom odsječku.

1.3. Utvrdite pomoću sustava NuSMV je li ispunjeno obilježje sigurnosti. Objasnite rezultat na temelju koda primjera (ne na temelju ispisa traga).

1.4. Specificirajte i napišite u CTL notaciji obilježje (engl. *liveness property*):

«Ako proces pokuša ući u kritični odsječak, konačno će i ući»

Specifikaciju napišite za oba procesa.

1.5. Utvrdite je li zadovoljeno obilježje životnosti. Koje sve probleme uočavate s ovom implementacijom?

1.6. U mutex_1ex.smv dodajte ograničenje pravednosti (engl. *fairness*): svaka instanca procesa obavlja se beskonačno mnogo puta. Napišite ovdje kako ono glasi.

1.7. Ponovno provjerite obilježje životnosti. Što smo postigli s ovim ograničenjem pravednosti, a što je još ostalo kao problem?

1.8. U mutex_1ex.smv dalje dodajte ograničenje pravednosti: svaka instanca procesa ne može beskonačno dugo ostati u **kritičnom** odsječku. Napišite ovdje kako ono glasi. Provjerite sad svojstvo životnosti za mutex_1ex.smv. Što se dogodilo?

1.9. U mutex_1ex.smv dodajte još jedno ograničenje pravednosti: svaka instanca procesa ne može beskonačno dugo ostati u **nekritičnom** odsječku. Napišite ovdje kako ono glasi. Provjerite sad svojstvo životnosti za mutex_1ex.smv. Što se dogodilo?

1.10. Specificirajte i napišite u CTL notaciji:

«Ako proces proc0 uđe u kritični odsječak, proc0 neće ponovo ući u kritični odsječak sve dok proc1 nije prošao kroz svoj kritični odsječak.»

1.11. Utvrdite je li zadovoljeno navedeno obilježje iz zadatka 1.10 za mutex_1ex.smv (uz ograničenja pravednosti). Koja obilježja protokola međusobnog isključivanja rješavaju ograničenja pravednosti prethodno navedena, a koji problem je još uvijek prisutan?

2. dio

2.1. Proučite primjer **mutex_2ex.smv**.

2.2. Je li zadovoljeno obilježje sigurnosti (1.dio, 2. pitanje)?

2.3. Je li zadovoljeno obilježje životnosti (1. dio, 4. pitanje)?

2.4. Dodajte sad ograničenja pravednosti kao kod zadataka 1.6, 1.8 i 1.9. Je li sad zadovoljeno obilježje životnosti?

2.5. Koji je problem u ovoj implementaciji međusobnog isključivanja (bez obzira na uključena ograničenja pravednosti)? Gdje sustav može «zapeti»? Problem specificirajte u CTL notaciji i provjerite pomoću sustava NuSMV.

2.6. Proučite primjer **mutex_3ex.smv**
Ovo je primjer uspješne implementacije međusobnog isključivanja. Zasniva se na rješenju kojeg je predložio T. Dekker a opisao E. W. Dijkstra.

2.7. Provjerite svojstva sigurnosti i životnosti. Jesu li zadovoljena (uz dodavanje tri ograničenja pristranosti iz 1. dijela)?

2.8. Koje se ideje za kontrolu pristupa kritičnom odsječku iz prethodnih (neuspješnih) pokušaja nameću u ovom rješenju?

3. dio

Proučite potpoglavlja 3.1, 3.2, 3.5 i 3.7 iz NuSMV priručnika "NuSMV 2.6 User Manual". Nakon toga riješite sljedeće zadatke:

3.1. Pokrenite interaktivno ljusku NuSMV-a. Učitajte model zadan datotekom **mutex_1ex_int.smv**.

3.2. Inicijalizirajte sustav za verifikaciju. Ukratko obrazložite što se sve događa prilikom pokretanja naredbe "go".

3.3. Simulirajte kretanje kroz 3 stanja (od proizvoljno odabranog početnoga). Navedite dvije naredbe koje se koriste da bi se to ostvarilo. Koju naredbu treba koristiti da bi se ispisao trag prolaska kroz ta stanja?

3.4. Provjerite stroj s konačnim brojem stanja. Kakva je relacija prijelaza tog automata? Može li doći do potpunog zastoja?

3.5. Koliko ukupno postoji stanja u modelu, a koliko postoji dosezljivih (engl. *reachable*) stanja? (napomena: *diameter* - promjer FSM-a je minimalan broj koraka potrebnih da bi se došlo do svih dosezljivih stanja)

3.6. Provjerite prvu po redu CTL specifikaciju (redni broj 0). Je li ona istinita ili lažna? Koje obilježje protokola međusobnog isključivanja se njome provjerava? Je li to obilježje zadovoljeno?

3.7. Provjerite drugu po redu CTL specifikaciju (redni broj 1). Je li ona istinita ili lažna? Koje obilježje protokola međusobnog isključivanja se njome provjerava? Je li to obilježje zadovoljeno?

3.8. Provjerite naredbe COMPUTE sustavu NuSMV (prva COMPUTE naredba ima redni broj 2 u modelu, a druga redni broj 3). Navedite rezultat izvođenja tih dviju naredbi. Uzima li naredba COMPUTE u obzir navedena ograničenja pravednosti?

4. dio

4.1. Proučite primjer **ferryman.smv**.

4.2. Specificirajte i napišite u CTL notaciji obilježje:

«Ne postoji siguran put kojim se dolazi do cilja problema.»

Pritom se u specifikaciji trebaju koristiti već definirane makro-instrukcije programa.

4.3. Provjerite zadano svojstvo. Je li ono zadovoljeno? Što nam u ovom slučaju daje ispis traga programa? Opišite redoslijed izvođenja kojim se uspješno dolazi do cilja problema.

4.4. Proučite primjer **tic_tac_toe.smv**.

4.5. Specificirajte i napišite u CTL notaciji sljedeća obilježja:

a. «Igrač 2 nema strategiju za pobjedu.»

b. «Igrač 2 ima strategiju da ne izgubi.»

c. «Igrač 1 ima strategiju da ishod ne bude izjednačeno.»

d. «Igrač 1 nema strategiju da ishod bude izjednačeno.»

Pritom uzmite u obzir da od početka igra do kraja igre treba biti odigrano točno 9 poteza.

4.6. Provjerite zadana svojstva. Koja od njih su istinita, a koja lažna?

4.7. Zadani kôd u NuSMV-u sadrži implicitni nedeterminizam uzrokovan varijablom *request*. Izmijenite zadani kôd tako da sadrži **isključivo** eksplicitni nedeterminizam.

```
MODULE main
VAR
      request: boolean
      flag: {red, blue};
ASSIGN
      init(flag) := red;
      next(flag) := case
                        request = TRUE : blue;
                        TRUE : red;
      esac;
```

4.8. Za zadani kôd u NuSMV-u nacrtajte odgovarajuću Kripke strukturu i odredite:
a) skup svih mogućih stanja – S_A
b) skup svih dosezljivih stanja – S_R (uz pretpostavku da su sva početna stanja dosezljiva)

```
MODULE main
VAR
      request : boolean;
      status : {ready, busy};
      negReq : boolean;
ASSIGN
      init(request) := FALSE;
      init(status) := busy;
      init(negReq) := FALSE;
      next(request) := case
                        (status = busy) : FALSE;
                        TRUE: TRUE;
      esac;
      next(status) := case
                        request : busy;
                        TRUE : ready;
      esac;
      next(negReq) := !request;
```